|  |  |
| --- | --- |
|  | МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ТАТАРСТАН  **Государственное автономное профессиональное образовательное учреждение**  **«ЗЕЛЕНОДОЛЬСКИЙ МЕХАНИЧЕСКИЙ КОЛЛЕДЖ»** (ГАПОУ «ЗМК») |

**09.02.07 «Информационные системы и программирование»**

**МДК 07.01. УПРАВЛЕНИЕ И АВТОМАТИЗАЦИЯ БАЗ ДАННЫХ**

**Отчет о практических работах**

**Исполнитель**: Карпов Евгений Степанович

**Группа**: 217

**Преподаватель:** Алемасов Евгений Павлович

**Дата сдачи** 06.05.2024 **Оценка** \_\_\_\_\_\_\_\_\_\_\_\_

**Подпись преподавателя**  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ЗЕЛЕНОДОЛЬСК – 2023**

**Работа No6. Создание базы данных в SQLite.**

**Реализовать БД:**

**Реализовать в Python функционал:**

**1 – добавление данных**

**2 – Вывод всех таблиц и возможность выбора таблицы. Затем вывод всех**

**данных из данной таблицы.**

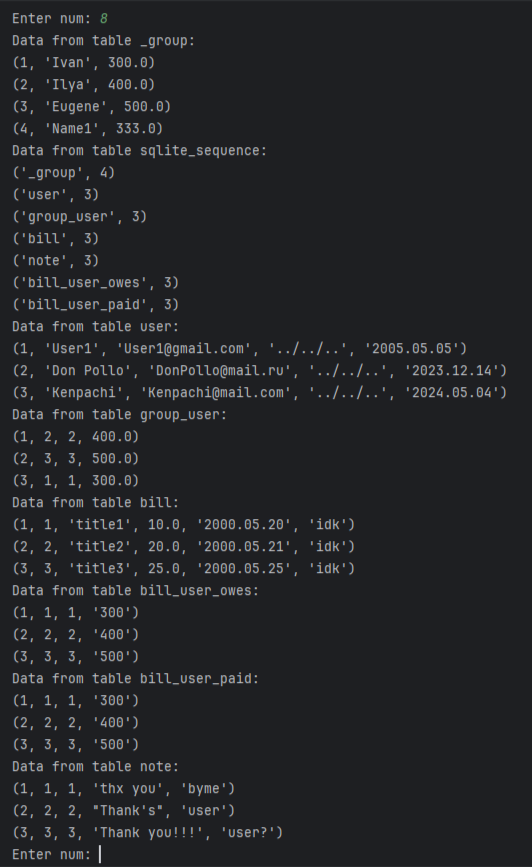
**Код программы:**

import sqlite3  
  
connection = sqlite3.connect("KarpovE217.db")  
cursor = connection.cursor()  
  
  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS \_group(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 name TEXT,  
 balance REAL  
 )''')  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS user(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 name TEXT,  
 email TEXT,  
 avatar TEXT,  
 member\_since DATE  
 )''')  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS group\_user(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 id\_group INTEGER,  
 id\_user INTEGER,  
 balance REAL,  
 FOREIGN KEY(id\_group) REFERENCES \_group(id),  
 FOREIGN KEY (id\_user) REFERENCES user(id)  
 )''')  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS bill(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 id\_group INTEGER,  
 title TEXT,  
 amount REAL,  
 date DATE,  
 created\_by TEXT,  
 FOREIGN KEY (id\_group) REFERENCES \_group(id)  
 )''')  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS bill\_user\_owes(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 id\_bill INTEGER,  
 id\_user INTEGER,  
 owes TEXT,  
 FOREIGN KEY(id\_bill) REFERENCES bill(id),  
 FOREIGN KEY (id\_user) REFERENCES user(id)  
 )''')  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS bill\_user\_paid(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 id\_bill INTEGER,  
 id\_user INTEGER,  
 paid TEXT,  
 FOREIGN KEY(id\_bill) REFERENCES bill(id),  
 FOREIGN KEY (id\_user) REFERENCES user(id)  
 )''')  
cursor.execute(  
 '''CREATE TABLE IF NOT EXISTS note(  
 id INTEGER PRIMARY KEY AUTOINCREMENT,  
 id\_bill INTEGER,  
 id\_user INTEGER,  
 message TEXT,  
 created TEXT,  
 FOREIGN KEY(id\_bill) REFERENCES bill(id),  
 FOREIGN KEY (id\_user) REFERENCES user(id)  
 )''')  
flag = 1  
menu = 10  
print("choose table \n 1: group \n 2: user \n 3: group\_user \n 4: bill \n 5: bill\_user\_owes \n 6: bill\_user\_paid \n"  
 " 7: note \n 8: all data \n for separate table : 1+1 = 11. \n 0: exit")  
while flag == 1:  
 menu = int(input("Enter num: "))  
 if menu == 0:  
 break  
 elif menu == 1:  
 name = input("Enter name: ")  
 balance = float(input("Enter balance: "))  
 cursor.execute('''INSERT INTO \_group (name, balance)   
 VALUES (?, ?)''', (name, balance))  
 elif menu == 2:  
 name = input("Enter name: ")  
 email = input("Enter email: ")  
 avatar = input("Enter avatar: ")  
 member\_since = input("Enter member\_since: ")  
 cursor.execute('''INSERT INTO user (name, email, avatar, member\_since)   
 VALUES (?, ?, ?, ?)''', (name, email, avatar, member\_since))  
 elif menu == 3:  
 id\_group = input("Enter id\_group: ")  
 id\_user = input("Enter id\_user: ")  
 balance = float(input("Enter balance: "))  
 cursor.execute('''INSERT INTO group\_user (id\_group, id\_user, balance)   
 VALUES (?, ?, ?)''', (id\_group, id\_user, balance))  
 elif menu == 4:  
 id\_group = input("Enter id\_group: ")  
 title = input("Enter title: ")  
 amount = input("Enter amount: ")  
 date = input("Enter date: ")  
 created\_by = input("Enter created\_by: ")  
 cursor.execute('''INSERT INTO bill (id\_group, title, amount, date, created\_by)   
 VALUES (?, ?, ?, ?, ?)''', (id\_group, title, amount, date, created\_by))  
 elif menu == 5:  
 id\_bill = input("Enter id\_bill: ")  
 id\_user = input("Enter id\_user: ")  
 owes = input("Enter owes: ")  
 cursor.execute('''INSERT INTO bill\_user\_owes (id\_bill, id\_user, owes)   
 VALUES (?, ?, ?)''', (id\_bill, id\_user, owes))  
 elif menu == 6:  
 id\_bill = input("Enter id\_bill: ")  
 id\_user = input("Enter id\_user: ")  
 paid = input("Enter paid: ")  
 cursor.execute('''INSERT INTO bill\_user\_paid (id\_bill, id\_user, paid)   
 VALUES (?, ?, ?)''', (id\_bill, id\_user, paid))  
 elif menu == 7:  
 id\_bill = input("Enter id\_bill: ")  
 id\_user = input("Enter id\_user: ")  
 message = input("Enter message: ")  
 created = input("Enter created: ")  
 cursor.execute('''INSERT INTO note (id\_bill, id\_user, message, created)   
 VALUES (?, ?, ?, ?)''', (id\_bill, id\_user, message, created))  
 elif menu == 8:  
 cursor.execute("SELECT name FROM sqlite\_master WHERE type='table';")  
 tables = cursor.fetchall()  
 for table in tables:  
 table\_name = table[0]  
 print(f"Data from table {table\_name}:")  
 cursor.execute(f"SELECT \* FROM {table\_name};")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 11:  
 cursor.execute("SELECT \* FROM \_group")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 22:  
 cursor.execute("SELECT \* FROM user")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 33:  
 cursor.execute("SELECT \* FROM group\_user")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 44:  
 cursor.execute("SELECT \* FROM bill")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 55:  
 cursor.execute("SELECT \* FROM bill\_user\_owes")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 66:  
 cursor.execute("SELECT \* FROM bill\_user\_paid")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 elif menu == 77:  
 cursor.execute("SELECT \* FROM note")  
 data = cursor.fetchall()  
 for row in data:  
 print(row)  
 else:  
 print("n/a")  
  
  
connection.commit()  
connection.close()

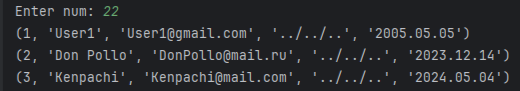
**Добавление данных:**



**Вывод всех данных:**



**Вывод отдельной таблицы:**



**P.S. Чтобы вывести данные из таблиц с номерами от 1 до 7, нужно дважды указать номер таблицы. Например, для таблицы 1  нужно написать 11, и так далее**